A neuro-fuzzy monitoring system. Application to flexible production systems.

Nicolas Palluat, Daniel Racoceanu, Noureddine Zerhouni

A neuro-fuzzy monitoring system. Application to flexible production systems.

N. Palluat, D. Racoceanu, N. Zerhouni

Laboratoire d'Automatique de Besançon, UMR CNRS 6596, 24 rue Alain Savary, 25000 Besançon, France.

Abstract

The multiple reconfiguration and the complexity of the modern production system lead to design intelligent monitoring aid systems. Accordingly, the use of neuro-fuzzy technics seems very promising. In this paper, we propose a new monitoring aid system composed by a dynamic neural network detection tool and a neuro-fuzzy diagnosis tool. Learning capabilities due to the neural structure permit us to update the monitoring aid system. The neuro-fuzzy network provides an abductive diagnosis. Moreover it takes into account the uncertainties on the maintenance knowledge by giving a fuzzy characterization of each cause. At the end, we illustrate the industrial usefulness of the proposed dynamic neuro-fuzzy monitoring system trough a flexible production system monitoring application.

Key words: UML, neural network, neuro-fuzzy, diagnosis, monitoring, maintenance, SCADA, CMMS, FMECA, fault tree.
PACS: 07.05.Mh, 87.57.Ra

1 Introduction

The improvement of the complexity of real production systems in a hard concurrent marketing context encourages the managers to give more importance to the maintenance functions. The industrial monitoring, which is one of the most significant of them, is divided into two tasks: the failure detection, and the failure diagnosis (failure localization and failure causes identification) (Pencolé, 2002; Tromp, 2000; Wan et al., 1999). More the system is complex, more the monitoring is difficult. An efficient monitoring system must be easy

Email address: {npalluat,daniel.racoceanu,zerhouni}@ens2m.fr (N. Palluat, D. Racoceanu, N. Zerhouni).

to improve due to system reconfiguration and experts / operators experiences feedbacks.

The heterogeneity of maintenance and production information is taken into account for the creation of our monitoring system. These information can be provided by:

- Failure Modes Effects and Critical Analysis - FMECA,
- Fault Tree - FT,
- Functional analysis,
- Production and maintenance operators and managers experiences,
- Computerized Maintenance Management System - CMMS,
- Supervisory Control And Data Acquisition - SCADA,
- ...

The proposed method concern all phases of the monitoring function: the fault detection and the fault diagnosis.

- *Dynamic detection tool*. The input of the detection system is given by sensors data. These data are treated dynamically. The output gives the operating mode (symptom) of the supervised system.
- *Diagnosis tool*. The input of the diagnosis system will be the degree of membership of each operating mode given by the detection. We find also external qualitative or quantitative inputs such as information given by operators to improve diagnosis. The output gives a list of possible causes ordered by credibility degree [1], and as complementary information: the severity degree. These degrees help the maintenance manager to evaluate and to plan the maintenance actions.

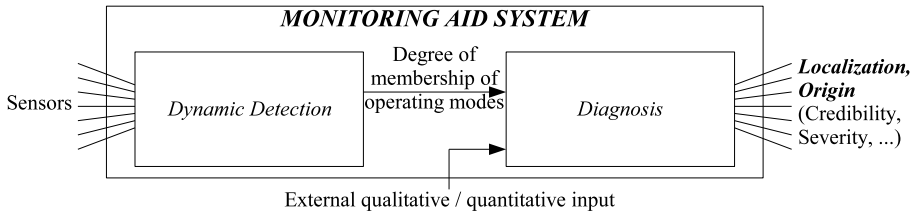


Fig. 1. Overview of the monitoring aid system.

During the process, the dynamic detection tool scans continuously the system. When a failure or a degradation occurs, an alarm is raised and the diagnosis tool starts. According to the information provided by the detection tool, the diagnosis tool proposes to the operator the possible causes of the symptom as well as the fuzzy interpretation of these causes. This point of view enables us to predict a possible failure.

[1] Credibility value / degree = membership degree of a variable to a membership function. In our work, the membership degree can be seen as a belief indicator

Our design approach follows the Unified Modelling Language (UML) (Larman, 2002; Rumbaugh et al., 1998). Several reasons led us to this choice (Morley et al., 2003):

(1) UML is a language normalized by OMG [2] (OMG, 2003) and UML specifications are free access.
(2) Interest shown by computer specialists for this modelling language.
(3) Possibility to use the same language since the requirements expression until the application generation.
(4) Ability to use the object oriented concepts to improve the design approach.

The requirements of the monitoring aid system are:

(1) ability to use an incomplete database and knowledge base;
(2) possibility to take into account new knowledge;
(3) ability to identify false alarms;
(4) easy to use;
(5) interfacing with industrial tools for maintenance management (FMECA, Fault Tree, CMMS);
(6) interfacing with industrial data acquisition tools (SCADA - Supervisory Control And Data Acquisition, Ethernet ...);
(7) integration of the tool in an industrial platform.
(8) possibility to interface with HMI (Human-Machine Interface) on PDA, laptop ...;

The paper is structured in four parts. The following paragraph presents the UML method to design our tool, three important use cases are developed. In a second time, we present important criteria to choose detection and diagnosis tools. In a third part, we briefly describe the two tools. At last, an industrial problem will be treated.

2 UML specification of the monitoring aid system

UML is articulated around several types of diagrams, each one of them being dedicated to the representation of the particular concepts of a software system. Our study focuses on use case . In order to develop the computing code of the monitoring aid system from the user requirements, we use the method given in Roques (2002).

[2] OMG web site: http://www.omg.org/,
OMG web site about UML: http://www.uml.org/

2.1 Use cases

Use cases enable to define the limits of the system and its relations with the environment. A use case is a specific manner to use the system. To lead to the use cases, we follow the next modelling approach:

- Identify the actors;
- Identify the use cases;
- Structure the use cases in packages;

2.1.1 Identification of the monitoring aid system actors

In our case, human actors are:

- Maintenance Manager;
- Tool Expert;
- Maintenance Operator.

We also take into account non-human actors[3] :

- SCADA - Supervisory Control And Data Acquisition;
- FT - Fault Tree;
- CMMS - Computerized Maintenance Management System;
- FMECA - Failure Modes Effects and Critical Analysis.

2.1.2 Identification of the use cases

We define a set of use cases that corresponds in different ways according to the actors that interacts with the monitoring aid system.

For the Maintenance Manager, we can define a set of four use cases:

- *create a new tool*, which allows to send to the Tool Expert all necessary data for the creation of the tool (configuration and initialization);
- *update the configuration of the tool*, allowing the Maintenance Manager to add a sensor to the detection tool;
- *update the model of the tool*, updating the settings of the tool when new maintenance information come from the CMMS;
- *raise an alert*, occurring when the monitoring aid system detects or predicts a failure.

[3] to simplify using UML, we do not make the distinction between methods, tools and systems; all of them are informational actors.

For the Maintenance Operator, we have only one use case:

- *ask for a diagnosis aid*, which allows to the Maintenance Operator to ask an assistance to the monitoring aid system.

At last, for the Tool Expert, we have four use cases:

- *create a new tool*, already presented in Maintenance Manager use cases;
- *configure the tool*, which allows to the Tool Expert to configure the monitoring aid system with the data extracted from the SCADA and the FT;
- *initialize the tool*, which allows to the Tool Expert to initialize the configured monitoring aid system with the data acquired from the FMECA and the CMMS, and to launch the data acquisition from the SCADA;
- *update the configuration of the tool*, already presented in Maintenance Manager use cases.

2.1.3 Packages

To improve our system, we will organize the use cases and group them in coherent functional sets. We thus create three packages:

- *Actors package*, set of all actors;
- *Off-line package*, set of use cases used when the monitoring aid system does not work : Create a new tool, Configure the tool, and Initialize the tool;
- *On-line package*, set of use cases when the monitoring aid system is used: Update the configuration, Update the model, Raise an alert and Ask for a diagnosis aid.

Three use cases have "include links":

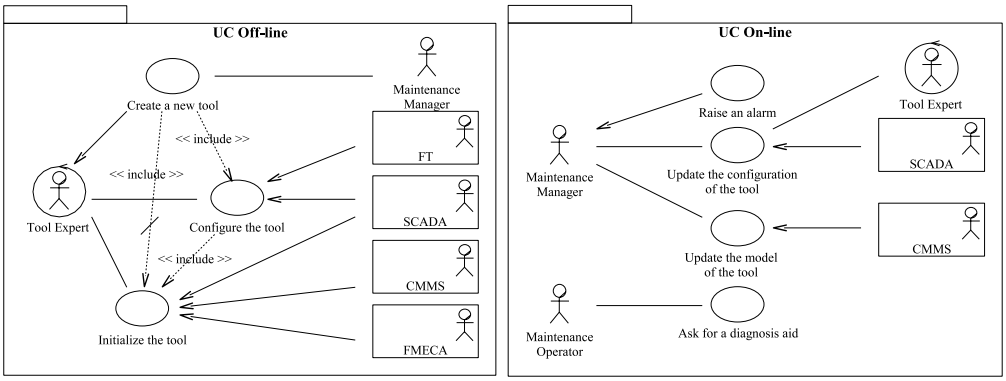
- between the creation of the new tool and the configuration.
- between the configuration of the tool and the initialization.
- between the creation of the new tool and the initialization.

We regroup all informations [4] in figures 2(a) and 2(b).

2.2 Detailed specification of the requirements

We specify the use cases, established previously, in a detailed textual description inspired by (Cockburn, 2000).

[4] We show the Actors inside the Off-line and On-line package to simplify the comprehension of the figures.



(a) Off-line package (b) On-line package

Fig. 2. Packages

We present three important use cases: Configure the tool, Initialize the tool and Ask for a diagnosis aid. These use cases define the principal characteristics of our monitoring aid system.

2.2.1 Configure the tool

We assume that as hardware as software are installed and functional and the Maintenance Manager knows the system. Main Scenario of the diagnosis tool configuration starts with the request of the Fault Tree. The Tool Expert translates the Fault Tree into a diagnosis tool. The architecture of the tool will also take into account the sensors availability through the SCADA.

Main Actor: Tool Expert.
Secondary Actors: The systems FT and SCADA.
Goal: Configuration of the tool.
Precondition: The maintenance manager asks to the tool expert to create a new tool.
Postcondition: The tool is configured.
Main scenario:

(1) FT actor provides the fault tree to the Tool Expert;
(2) the Tool Expert configures the diagnosis tool with the fault tree;
(3) the Tool Expert configures the detection tool i.e. subscription to SCADA in order to access information from useful sensors.

2.2.2 Initialize the tool

The monitoring aid system initialization consists in setting values that were extracted from the existent FMECA and CMMS. This process will take into account the criticity (frequency, severity) of the failures and the associated

Symptom and Origin given by the CMMS.

Main Actor: Tool Expert.
Secondary Actors: The systems FMECA, CMMS and SCADA.
Goal: Initialization of the tool.
Precondition: The tool is configured.
Postcondition: The tool is initialized.
Main scenario:

(1) FMECA actor provides the FMECA to the Tool Expert;
(2) the Tool Expert analyzes the FMECA and extracts useful data (Operating modes, causes, frequencies and severities);
(3) the Tool Expert initializes the detection and diagnosis tools with the extracted data;
(4) the Tool Expert initializes the diagnosis tool i.e. subscription to maintenance events of the CMMS.
(5) detection tool receives sensors information from SCADA.

2.2.3 Ask for a diagnosis aid

A failure has been detected or predicted and/or the Maintenance Operator needs a diagnosis aid. The starting point of the diagnosis is the Maintenance Operator assistance request. The system will suggest then a diagnosis and, finally, the Maintenance Operator will validate the diagnosis.

Main Actor: Maintenance Operator.
Goal: The tool gives a diagnosis aid.
Precondition: The tool is working (Configured and Initialized). A failure was detected or predicted and/or the Maintenance Operator needs the assistance of the tool.
Main scenario:

(1) Maintenance Operator requests a diagnosis aid;
(2) the system provides a set of possible causes classified by credibility and severity degrees;
(3) Maintenance Operator validates the diagnosis.

Exceptions:

1a. Maintenance Operator chooses a detailed diagnosis.
(1) Maintenance Operator reaches a specialized form allowing him to add information which cannot be given by the detection tool (smoked, odors ...) and the use case continue to the step 2 of the main scenario.
3a. Maintenance Operator is not satisfied with the results.
(1) Maintenance Operator returns to the step 1 of the main scenario to launch a new request.
(1) Maintenance Operator gives up the request. The use case finishes (failure).

2.3 Synthesis of UML on the monitoring aid system

We present the seven use cases of the monitoring aid system and we describe three of them. These descriptions will allow us:

- to determine the optimized tools for the requirements presented earlier, and
- to specify how to use the monitoring aid system.

3 Choice of detection and diagnosis tools

Using the requirements defined before, we can specify tools answering to these specifications.

Monitoring methodologies are based on two concepts (Dash and Venkatasubramanian, 2000):

- monitoring methods with process model and
- monitoring methods without model.

The model of a system is generally difficult to obtain, especially for complex systems reconfigurable or subjected to hazard. For flexibility and adaptability reasons, we choose the class of monitoring methods without model and more precisely using artificial intelligence technics.

3.1 Choice of the detection tool

Among AI technics, our study focuses on the use of pattern recognition for the detection with the artificial neural networks. This tool gives interesting results thanks to:

- their training and parallel computation capabilities,
- their capacity to solve problems inherent to the system non-linearity,
- their computation speed when implemented in hardware.

The training phase uses a data base obtained from industrial maintenance tools (SCADA, FMECA ...) able to give sensors information and associated operating modes. The use of neural networks for industrial applications such as dynamic detection requires to take into account the temporal aspect. Accordingly, we have done a state of art of the temporal neural networks (Palluat et al., 2005). The conclusion of this state of art permit us to choose the Recurrent Radial Basis Function Neural Network as the most efficient detection

tool.

3.2 Choice of the diagnosis tool

Diagnosis takes into account numeric and symbolic data. Moreover, diagnosis needs causal knowledge on the system. Indeed, a failure is described by the relation between its causes and its effects. Thus, diagnosis problem consists of finding explanations from the observed symptoms. The inference that allows to "go back to the causes" is called Abductive Inference (Monnin et al., 2004; Peng and Reggia, 1990).

Given the complexity of diagnosis problems, many methods have been developed using different AI tools. In (Monnin et al., 2004), the authors give a classification of these methods by kind of diagnosis (based on behavioral models, recognition methods, based on explicative models). They determined four important criteria to resume the properties of a diagnosis tool:

(1) The model acquisition is the first step to make a diagnosis tool. Thus, it is necessary to have knowledge on system to diagnose. The explicative models seem to be the best adapted to express the causal knowledge on a system, which are essential to carry out a diagnosis.
(2) These knowledge based on human expertise are uncertain. Fuzzy logic is the best tool to express and take into account these uncertainties.
(3) Moreover, a diagnosis tool has to be robust and relatively generic.
(4) At last whatever tool we use, results have to be validated by an expert.

Taking into account these requirements for the diagnosis, we focus on methods based on explicative models. They are well adapted for the modelling of the causal relations essential to the diagnosis.

In our study, we consider a neural network to model relations between failures/degradations and causes. Moreover, in order to take into account uncertainties linked to the maintenance knowledge, we introduce fuzzy logic, able to introduce fuzzy degrees of credibility associated to the failures. Finally, we use an abductive approach that characterizes the causes starting from the observations. The algorithm corresponds to a downward approach in the fault tree. Once the top event exists with a certain fuzzy degree of credibility (the observation is quantitative), it is propagated in the net. So, each fault which could be at the origin of the top event is characterized by its own fuzzy degree.

4 Description of the monitoring aid system

4.1 Configuration of the monitoring aid system

4.1.1 Configuration of the detection tool

The detection tool is based on the recurrent RBF neural network (Zemouri et al., 2001, 2003). The RRBF neural network considers the time as an internal representation (Chappelier and Grumbach, 1998; Elman, 1990; Zemouri et al., 2003). The dynamic aspect is obtained using an additional self-connection of input neurons with a sigmoid activation function. These looped neurons are a special case of the Locally Recurrent Globally Feedforward architecture, called local output feedback (Tsoi and Back, 1994). The RRBF neural network can thus memorize a part of the historic of the input signal. The following figure shows this architecture:

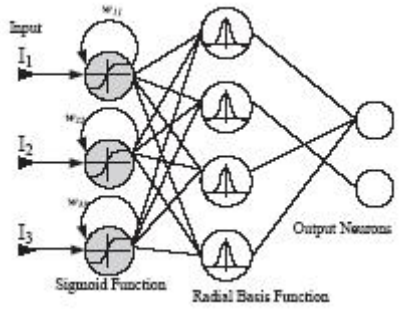


Fig. 3. RRBF neural network

Inputs are connected to the sensors information through SCADA. These information are normalized before entering in the neural network. The RRBF outputs will be defined in the Initialization phase.

The parameters of the input neurons are deducted from the dynamic of the sensor signal. The output of the neuron i is defined by the next function:

$$O_i(t) = \frac{1 - \exp\left(-k_i \cdot I_i(t) - k_i \cdot w_i \cdot O_i(t-1)\right)}{1 + \exp\left(-k_i \cdot I_i(t) - k_i \cdot w_i \cdot O_i(t-1)\right)} \tag{1}$$

with I_i the input, w_i the self-connection weight and k_i the parameter of the sigmoid.

The two parameters are obtained following the next algorithm:

Algorithm 1 *Calculate* $\gamma = f(T, I, C, N, S_o)$

$\{T$ *normalized training sequences*$\}$
$\{I$ *number of inputs*$\}$
$\{C$ *number of training sequences*$\}$
$\{N$ *number of vectors in a training sequence*$\}$
$\{S_o$ *forgetting threshold*$\}$
$\{$*Definition of the local variables*$\}$
γ, γ^{new}, γ_{max}, γ^{new}_{max}, γ_{min}, γ^{new}_{min}, O_{n-1}, O_n, Δ : ***real***;
$\{$*Initialization of the local variables*$\}$
$\gamma \leftarrow 1$;
$\gamma_{max} \leftarrow 2$;
$\gamma_{min} \leftarrow 0$;
$O_{n-1} \leftarrow 1$; $\{$*Initialization of the output*$\}$
$\Delta \leftarrow 0,01$; $\{$*Precision of* $\gamma\}$
$\{$*Calculate* $\gamma\}$
$\{\gamma$ *is the product* $k_i \cdot w_i$ *constant for all* $i\}$
repeat
  for $i = 1$ *to* N ***do***
    $O_n \leftarrow \dfrac{1 - \exp(-\gamma \cdot O_{n-1})}{1 + \exp(-\gamma \cdot O_{n-1})}$;
    $O_{n-1} \leftarrow O_n$;
  end for
  $\{$*adjustment of the values of* γ, γ_{max} *and* $\gamma_{min}\}$
  if $O_{n-1} \leq S_o$ ***then***
    $\gamma^{new}_{max} \leftarrow \gamma$;
    $\gamma^{new} \leftarrow \dfrac{\gamma_{max} + \gamma}{2}$;
    $\gamma \leftarrow \gamma^{new}$;
    $\gamma_{max} \leftarrow \gamma^{new}_{max}$;
  else
    $\gamma^{new}_{min} \leftarrow \gamma$;
    $\gamma^{new} \leftarrow \dfrac{\gamma + \gamma_{min}}{2}$;
    $\gamma \leftarrow \gamma^{new}$;
    $\gamma_{min} \leftarrow \gamma^{new}_{min}$;
  end if
until $\gamma_{max} - \gamma_{min} \leq \Delta$
for $i = 1$ *to* I ***do***
  $\{$*Calculate* $k_i\}$
  $k_i \leftarrow \dfrac{\ln\left(2 + \sqrt{3}\right)}{\max\limits_{j \in C, l = 1, \ldots N} \left|T^j_{i,l}\right|}$;
  $\{$*Calculate* $w_i\}$
  $w_i \leftarrow \dfrac{\gamma}{k_i}$;
end for
return k, w;

4.1.2 Configuration of the diagnosis tool

In the UML description, the transformation of the fault tree create the diagnosis tool. We have associated to each type of gate (in this paper, AND and OR gate) a neuro-fuzzy architecture. The figure 4 shows this transformation.

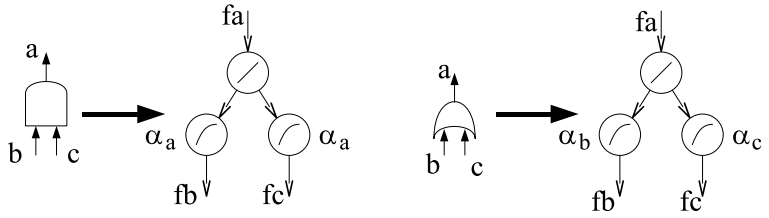


Fig. 4. Transformation of AND and OR gates into a neuro-fuzzy architecture

We can see in this figure two types of neurons with the following activation functions:

- linear activation function:

$$f(x, \alpha) = x, \ \forall x \in [0, 1] \tag{2}$$

- sigmoidal activation function:

$$f(x, \alpha) = \frac{1 - \exp(-\alpha \cdot x)}{1 + \exp(-\alpha \cdot x)}, \ \forall x \in [0, 1] \tag{3}$$

Concerning the basic events (according to the definition of a fault tree event), we introduce an additional transformation: each basic event is transformed into a neuron with linear activation function. We will see the reason of this transformation in the initialization phase.

4.2 Initialization of the monitoring aid system

4.2.1 Initialization of the detection tool

The RRBF neural network outputs are given by the failure modes of FMECA.

The learning algorithm of the detection tool is based on DDA (Dynamic Decay Adjustment) algorithm (Berthold and Diamond, 1995) which has been modified following three points:

- any prototype can validate a training pattern;

- a maximum standard deviation is added:

$$\sigma_{\max} \leftarrow \sqrt{\frac{-(0.1)^2}{2 \cdot \ln(\theta^+)}} \tag{4}$$

- conflicting prototypes will be adjusted only when there is an introduction of a new prototype.

The hidden layer is composed of units with a selective response for some range of the input variables. Each unit has an overall response function, a Gaussian:

$$R(x) = \exp\left(\frac{-\|x - r\|^2}{2 \cdot \sigma^2}\right) \tag{5}$$

Here x is the input, r is the center of the radial basis function (RBF) and σ determines its standard deviation.

Algorithm 2 *Modified DDA algorithm*

{θ^+ *and* θ^-*: two threshold to limit the activation between two conflicting classes.*}
{A *is the weight for each RBF*}
repeat
  {*Reset output weights*}
  for all *prototypes* i *of class* k p_i^k ***do***
    $A_i^k \leftarrow 0$;
  end for
  {*Train one complete epoch*}
  for all *training pattern* x *of class* c ***do***
    if $\exists p_i^k : R_i^k(x) \geq \theta^+$ ***then***
      $A_i^k \leftarrow A_i^k + 1$;
    else
      {*"Commit": Introduce new prototype*}
      add new prototype $p_{m_c+1}^c$ *with:*
      $r_{m_c+1}^c \leftarrow x$;
      {*Note that for the first neuron, the standard deviation will be:* $\sigma_{\max}$}
      $\sigma_{m_c+1}^c \leftarrow \min\left\{\sigma_{\max}, \max\limits_{k \neq c \wedge 1 \leqslant j \leqslant m_k}\left\{\sigma : R_{m_c+1}^c\left(r_j^k\right) \leqslant \theta^-\right\}\right\}$;
      $A_{m_c+1}^c \leftarrow 1$;
      $m_c \leftarrow m_c + 1$;
      {*"Shrink": Adjusting conflicting prototypes*}
      for all $k \neq c$, $1 \leqslant j \leqslant m_k$ ***do***
        $\sigma_j^k \leftarrow \min\left\{\sigma_{\max}, \max\left\{\sigma : R_j^k(x) < \theta^-\right\}\right\}$;
      end for
    end if

end for
until *no further introduction of new prototype*

4.2.2 Initialization of the diagnosis tool

In the UML description, the Tool Expert extracts data from FMECA and introduces these data in the diagnosis tool. All the frequencies in the FMECA are deducted by fuzzification from the MTBF. To calculate each α corresponding at each frequency, it is necessary to know the number of frequencies. So, we have s α $(\alpha_1 \dots \alpha_s)$. We introduce a new α: α_0 for causes which have never happened. To determine each α we consider the following hypothesis:

- α_0 is determined by:

$$f\left(0.5, \alpha_0\right) = 0.2 \tag{6}$$

- α_s is determined by:

$$f\left(0.5, \alpha_s\right) = 0.8 \tag{7}$$

- all α are linearly distributed.

Following these hypothesis, we obtained all α by:

$$\alpha_i = 2 \cdot \ln\left(\frac{6 \cdot s + 3 \cdot i}{4 \cdot s - 3 \cdot i}\right) \tag{8}$$

For the others α , we make an update of the neighborhood of the output neurons. In Looney and Liang (2002), the authors developed a method for updating the neighborhood of bayesian network. In this method, the fuzzy influence propagation is bidirectional. We use this notion of updating the neighborhood on our study. The neighbors of a neuron N are the neurons directly or not connected to it. The neighbors have two properties, their level and their family link (parents or children). A neighbor of level 1 of a neuron N is directly connected to this neuron N. A neighbor of level 2 is directly connected to a neighbor of a level 1. A parent is a neighbor with the connection established from him to the neuron N and a child is a neighbor with the connection from the neuron N.

We define some principles which enable the determination of all α:

(1) each neuron have three states (”not tested”, ”testing” and ”tested”);
(2) each neuron with no children is in state ”tested”, the others are in state ”not tested”;
(3) each neuron with no children have is α determined by equation 8;

(4) for all neighbor "not tested" of level 1 and parent of a neuron "tested", their states become "testing";
(5) for all neurons "testing", their α is determined by the maximum of the α of their child of level 1 and their states become "tested";
(6) for all neurons "tested", if there are two neurons with the same name of α [5], then all the corresponding α takes the maximum value of α with the same name.

In the example of the figure 5(b), the neurons N_6, N_{10} and N_{11} have no children and their states becomes "tested" (condition 2). The corresponding α are determined by the FMECA (condition 3). The states of neurons N_4, N_8, N_9 and N_2 become "testing" (condition 4). α_B of N_4 is equal to the α of N_6, α_E of N_8 is equal to the α of N_{10} and α_F of N_9 and N_2 is equal to the α of N_{11} and the states of these three neurons become "tested" (condition 5). The condition 6 is verified in case of α_F.

The advantage of this system is the link between the detection tool, RRBF neural network, and the diagnosis tool, the neuro-fuzzy network. To connect the two tools, we used the FMECA. The detection tool's outputs are the failure modes found in the FMECA. For each failure mode, we can find one or many causes. These causes are present in the fault tree. So, the link is on the state witch is the parent of all causes. For example, in the figure 5(a), if C and E are the causes of the failure mode M, then the link will be on B. All the output of the detection tool will be linked on the diagnosis tool. These links will be the inputs of the neural networks. We will modify the network to take this new information. The modification will modify the propagation sens of the information. So, all causes will have a degree of credibility for each ask for diagnosis aid. This modification of sens will affect also the neurons. We introduce a new activation function which is used for the modification of a neuron with sigmoïdal activation function. The new activation function is logarithmic and is defined by:

$$f(x,\alpha) = \min\left(1, \frac{-1}{\alpha} \cdot \ln\left(\frac{1-x}{1+x}\right)\right), \ \forall x \in [0,1] \tag{9}$$

To modify the network we establish principles based on the update of the neighborhood.

(1) each neuron have three states ("not tested", "testing" and "tested");
(2) all neurons are in state "not tested";
(3) each neuron linking with the detection tool or with an external event is a neuron with linear activation function and its state is "testing";
(4) all neighbor "not tested" of level 1 of a neuron "tested" become the child

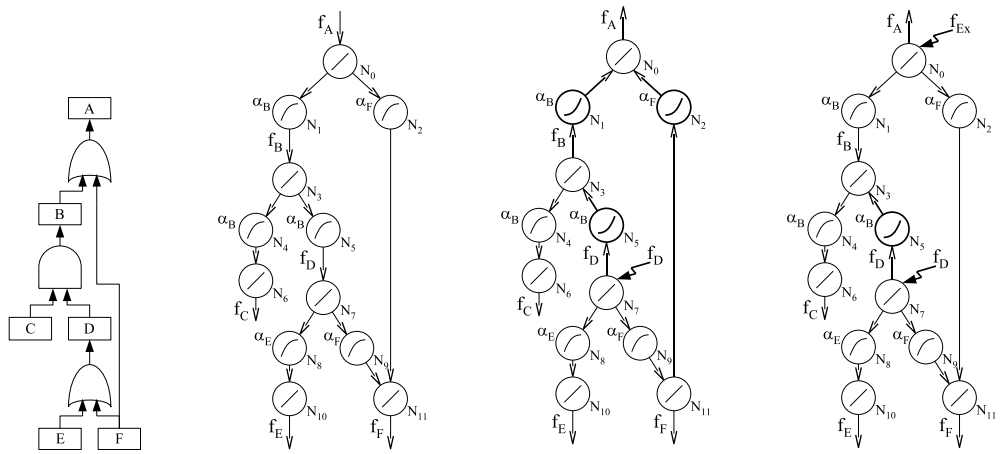
[5] it's the case for a AND gate, figure 4

of this last one (modification of propagation sens if necessary) and their states become "testing";

(5) for each neurons "testing", if all its parents are "tested" its state become "tested" else it is necessary to change the activation function[6] before it become "tested".

In the example of figure 5, the event D is linked to the detection tool by the degree f_D. The state of the neuron N_7 become "tested" (condition 3). The neurons N_5, N_8 and N_9 become "testing" and the propagation sens is inversed between N_7 and N_5 (condition 4). N_8 and N_9 become "tested" without modification, but N_5 change his activation function because his parent N_3 is "not tested" (condition 5). The modified network is on the figure 5(c).

In the figure 5(d), we add an external input that can be given by the operator for a detailed diagnosis.



(a) Fault Tree

(b) neuro-fuzzy network

(c) neuro-fuzzy network linked with detection tool

(d) neuro-fuzzy network linked with detection tool and with external input

Fig. 5. Example of transformation of the fault tree in a neuro-fuzzy network showing the link with the detection tool and the addition of an input

After configuration and initialization phases, the monitoring aid system is autonomous.

[6] the neurons with linear activation function will not change, but the neurons with sigmoïdal activation function become neurons with logarithmic activation function and *vice-versa*

5 Example

The proposed monitoring aid system was validated using a flexible production system, available to the "Institut de Productique"[7] of Besançon (France). This platform is equipped with several PLC communicating between them through a local industrial network. The flexible system permits to move pallets which can receive components to assembly.

The network allows to exchange the information received by the PLC concerning the changes of states of the sensors, the sequences of the control program, and operations made on the pallets.

The platform is divided into five stations. Each station has its own PLC. They work independently.

For this example, we limit the study to the input of one station, involving the actors SCADA, FT and FMECA. We can find below the extracted overview of this critical part (Fig. 6), the extracted fault tree (Fig. 7) and the extracted FMECA (Table 1).

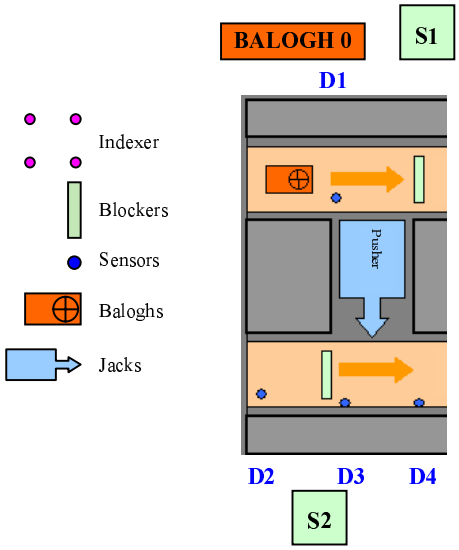


Fig. 6. Extracted overview of the critical part

We use the development software LabVIEW to implement our tool. We choose LabVIEW because his environment is built to combine signal acquisition, measurement analysis and data presentation, thus avoiding the intermediate storage of those. We implement our monitoring aid system in an industrial computer PXI proposed by National Instrument. The integration is thus simplified. The figure 8 shows the prototype of our monitoring aid system.

[7] Institut de Productique, Besançon, France http://www.institutdeproductique.com/

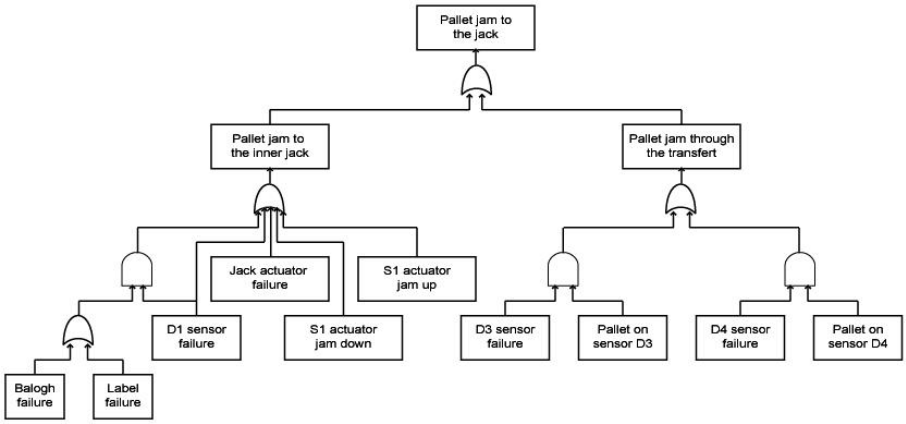


Fig. 7. Flexible manufacturing sub-system fault tree

Table 1
Flexible manufacturing sub-system extracted FMECA

Failure Modes	Cause	Frequency	Severity
Pallet jam to the inner jack	D1 sensor failure	4	2
Pallet jam to the inner jack	S1 actuator jam up	3	4
Pallet jam to the inner jack	Jack actuator failure	1	2

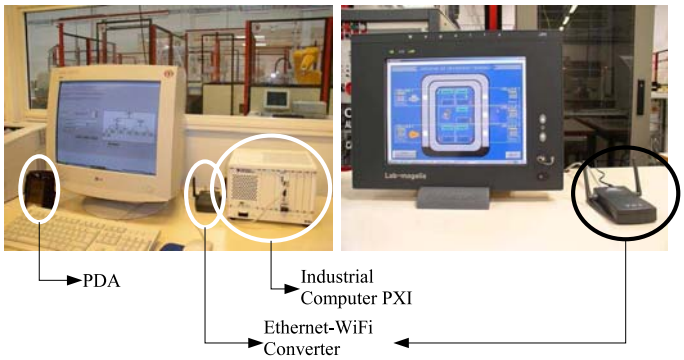


Fig. 8. Overview of the monitoring aid system prototype

Given the AND and OR influences and the link with the detection tool, we obtain the neuro-fuzzy network shown in figure 9. AND/OR dependencies are given with only two types of neurons with linear and sigmoidal activation. The link with the detection adds a new type of neurons with logarithmic activation.

We assume that the detection tool gives the failure mode "Pallet jam to the inner jack" with the membership degree 0.57. These information are treated by the diagnosis tool and the results are given in the figure 10.

This degree is propagated through the fuzzy neural network. The maintenance knowledge of the FMECA and the causal relations of the fault tree used in the fuzzy neural network allow to suspect the event corresponding to f_E:

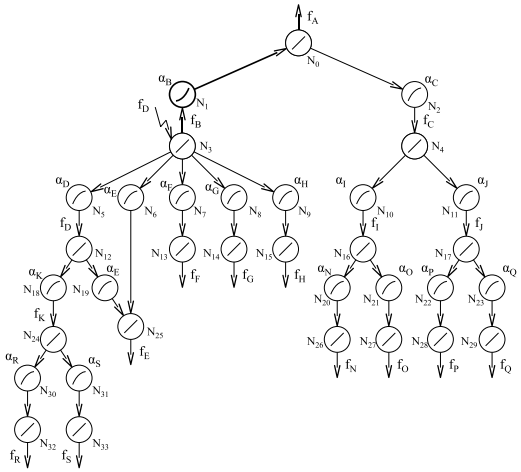


Fig. 9. Neuro-fuzzy network of specified zone aith link with the detection tool

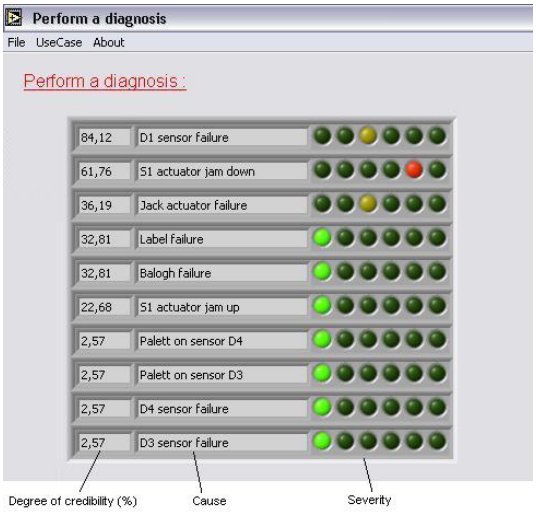


Fig. 10. Result of the diagnosis

"D1 sensor failure" as the fault origin with a credibility degree of 84.12%. This information is pertinent with the learning information. it aids the maintenance operator in his diagnosis.

No comparison with other approaches has been added in this article because we develop a method which represents both part of monitoring (detection and diagnosis) and we weren't able to identify a technique with the same property. We shall do a comparison for each part of the monitoring aid system. Concerning the detection tool, a comparative study has been made in Palluat et al. (2005). For the diagnosis tool, we can't find a benchmark for diagnosis. Diagnosis tools comparison is not easy because of the learning phase data, the inputs and the outputs of the different techniques are not always of the same type. For all these reasons, we make choice to not include a comparison in this article.

6 Conclusion

In this article, we present a new monitoring aid system designed using an UML approach. Our system is divided into two parts: a dynamic neural network detection tool and a neuro-fuzzy diagnosis tool.

Before using the neuro-fuzzy system, two steps are necessary: the first one is the configuration where data are collected and extracted to create the tools and the second one is the initialization where data are learned by the tools.

In use, the tools are in detection state where the dynamic neural network determines in which mode the system works, with an associated membership degree. When a failure or a degradation occurs, the detection tool raises an alert. When a diagnosis is requested, the diagnosis tool uses data from the detection tool to give possible locations and causes of the problem, classified by credibility and severity degrees. During the monitoring, the maintenance manager can improve the tools by configuration and/or model updating.

We illustrate the use of our monitoring aid system on an industrial flexible platform. No comparison with other approaches has been added in this article. The difficulties are to find a benchmark of diagnosis and to find monitoring tools.

The neuro-fuzzy network provides an abductive diagnosis. Moreover, it takes into account the uncertainties of the maintenance knowledge by giving a fuzzy characterization of each causes. So, location and identification of the fault causes are implicitly given by the events in the fault tree. Learning capabilities due to the neural structure permit us to update the configuration and the model, according to the new events given by the CMMS.

Further work will investigate methods to improve the on-line learning of the monitoring aid system. Some comparison with other approaches must be integrate.

References

Berthold, M. R., Diamond, J., 1995. Boosting the performance of rbf networks with dynamic decay adjustment. In: Tesauro, G., Touretzky, D. S., Leen, T. K. (Eds.), Advances in Neural Information Processing Systems. Vol. 7. MIT Press, Cambridge, MA, Ch. 5, pp. 521–528.

Chappelier, J.-C., Grumbach, A., May 1998. RST: a connectionist architecture to deal with spatiotemporal relationships. Neural Computation 10 (4), 883–902.

Cockburn, A., 2000. Writing Effective Use Cases. Addison-Wesley Pub Co.

Dash, S., Venkatasubramanian, V., 2000. Challenges in the industrial applications of fault diagnostic systems. In: Process Systems Engineering (PSE 2000). Vol. 24 of Computers and Chemical Engineering. pp. 785–791.

Elman, J. L., 1990. Finding structure in time. Cognitive Science 14 (2), 179–211.

Larman, C., 2002. Applying UML and patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process. Prentice Hall.

Looney, C. G., Liang, L. R., November 2002. Inference via fuzzy belief networks. In: the ISCA 15th International Conference on Computer Applications in Industry and Engineering (CAINE'2002). San Diego, CA, USA, pp. 25–28.

Monnin, M., Palluat, N., Racoceanu, D., Zerhouni, N., November 2004. Diagnosis methods using artificial intelligence. application of fuzzy petri nets and neuro-fuzzy systems. In: Third Conference on Management and Control of Production and Logistics, MCPL2004. Santiago de Chile, p. 6.

Morley, C., Hugues, J., Leblanc, B., 2003. UML pour l'analyse d'un système d'information, 2ème édition. Dunod.

OMG, March 2003. Unified Modeling Language Specification. Version 1.5. Object Management Group.

Palluat, N., Racoceanu, D., Zerhouni, N., 2005. Utilisation des réseaux de neurones temporels pour le pronostic et la surveillance dynamique: Etude comparative de trois réseaux de neurones récurrents. Revue d'Intelligence Artificielle, RSTI série RIA, 19 (6), 913–950.

Pencolé, Y., June 2002. Diagnostic décentralisé de systèmes à événements discrets : application aux réseaux de télécommunications. Ph.D. thesis, Université de Rennes I.

Peng, Y., Reggia, J. A., 1990. Abductive inference models for diagnostic problem-solving. Symbolic Computation. Springer-Verlag New York, Inc.

Roques, P., 2002. UML - Modéliser un site e-commerce. Les cahiers du programmeur. Eyrolles.

Rumbaugh, J., Jacobson, I., Booch, G., 1998. The unified modeling language reference manual. Addison-Wesley.

Tromp, L., January 2000. Surveillance et diagnostic de systèmes industriels complexes: une approche hybride numérique/symbolique. Ph.D. thesis, Université de Rennes I.

Tsoi, A. C., Back, A. D., 1994. Locally recurrent globally feedforward networks, a critical review of architectures. IEEE Transactions on Neural Networks 5 (2), 229–239.

Wan, Y. H., Marzuki, K., Syed, A. F. S. Z., 1999. Transformer fault diagnosis using fuzzy logic interpretations. In: Instrument Asia Technical Symposium'99. Singapore, p. 10.

Zemouri, R., Racoceanu, D., Zerhouni, N., October 2001. The RRBF: Dynamic representation of time in radial basis function network. In: 8th IEEE International Conference on Emerging Technologies and Factory Automa-

tion, ETFA'2001. Vol. 2. Antibes, Juan-les-Pins, pp. 737–740.
Zemouri, R., Racoceanu, D., Zerhouni, N., 2003. Recurrent radial basis function network for time-series prediction. Engineering Applications of Artificial Intelligence 16, 453–463.

Nicolas Palluat is a Ph.D. student in Automatics at Faculty of Sciences of the University of Franche-Comté, Besançon, France. He received his MS in Automatics, Productics and Informatics in 2002 from the same university. He obtained his BS in Mechanics and BS Automatics in 1998 and 2000, respectively. His research interests include artificial neural network, fuzzy logic and abductive diagnosis. He joined the Laboratory of Automatics of Besançon in 2001.